

Manipulating ephemeral data with git

Max Halford
Défi IA 2021
2021.10.07



There's an **API** for everything



Bike sharing stations



News articles



Stock markets



Flight schedules



Weather forecasts

Longitudinal studies

 Availability of bikes in time

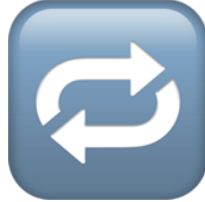
 News article trend analysis

 Stock market volatility analysis

 Evolution of plane ticket pricing

 Accuracy of weather forecasts

Accessing and storing temporal data

- If you're lucky, an API will provide historical data
- If not, you'll have to store it yourself:
 1. Create a database schema
 2. Write some code to fetch/scrape data
 3. Store the data in the database
 4. Rinse and repeat steps 2 and 3 

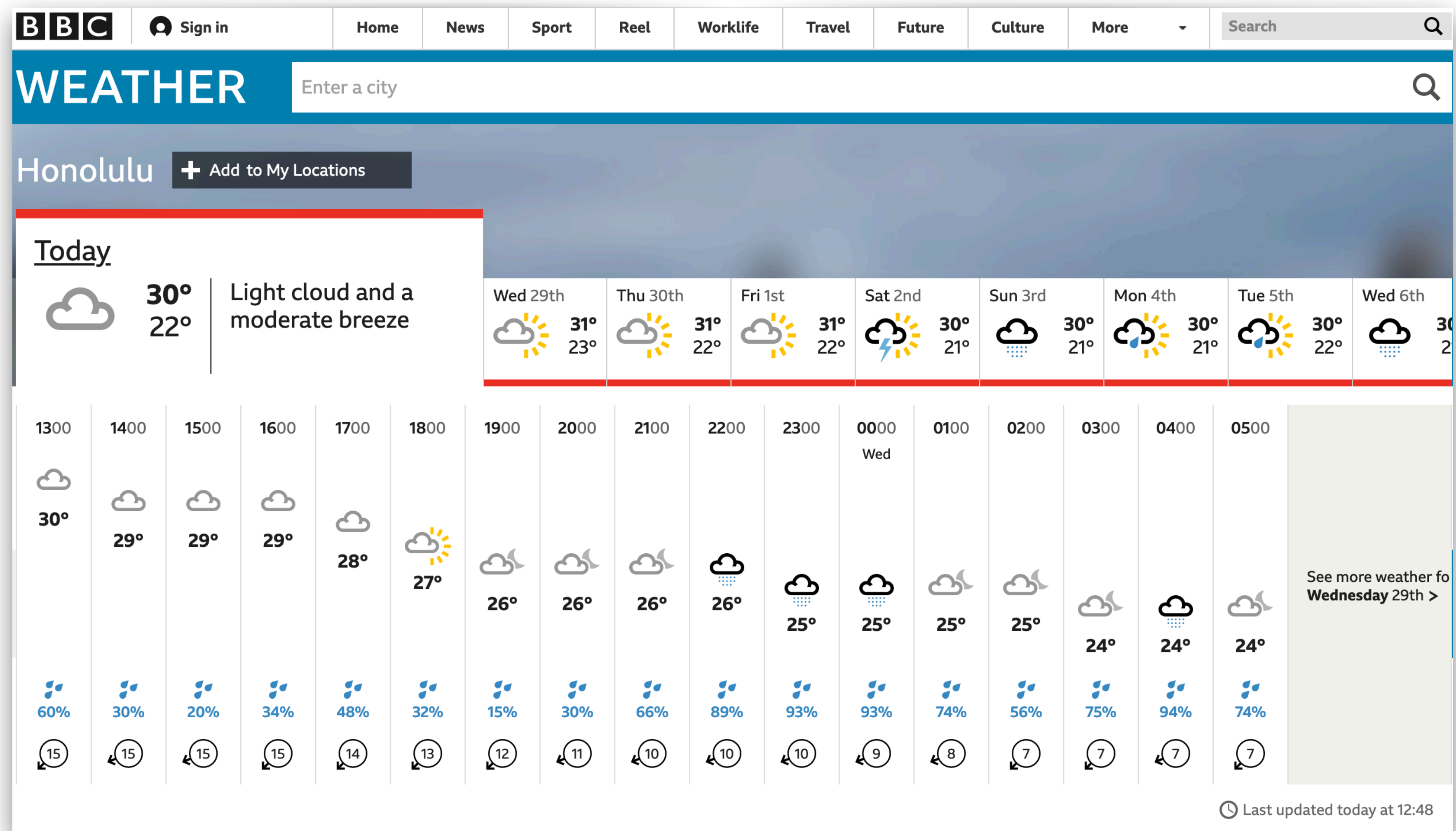
Friction of the tradition

- 😞 You have to create and maintain a database
- 😞 You have to write code to fetch/scrape data
- 😞 You have to host a periodic process
- 😞 It's 2021, all that is boring
- 🌟 We just want to analyse data

A different approach

1. Store raw data in a JSON file
2. Version it with git
3. Run this process periodically with GitHub Actions
4. Use git to time travel and analyse the data offline

Case example: the BBC Weather website



Find the API endpoint by inspecting traffic

The screenshot shows the Chrome DevTools Network tab with the 'Réseau' (Network) panel selected. The top toolbar includes icons for 'Inspecteur', 'Console', 'Débogueur', 'Réseau', and a red error icon with the number 7. Below the toolbar, there's a search bar 'Filtrer les URL' and a checkbox 'Désactiver le cache'. The main table lists network requests, with the first one highlighted: a GET request to 'weather-broker-cdn.api.bbc.co.uk/en/forecast/aggregated/5856195' returning a JSON response of 1,35 Ko. Below the table, a summary bar shows '58 requêtes', '4,64 Mo / 326,42 Ko transférés', and 'Terminé en : 8,60 s'. The bottom section shows the 'Réponse' (Response) for the selected request, displaying a JSON object with a 'features' array containing a 'Feature' object with a 'Point' geometry and a 'properties' object with weather observations.

État	Mét...	Domaine	Fichier	Initiateur	Type	Transfert	Ta
200	GET	weather-brok...	forecasts-observations?locations=...	5856195:1 (...)	json	1,35 Ko	13
304	GET	gn-flagpoles....	bbcdotcom	5856195:1 (...)	plain	mis en cache	88
304	GET	emp.bbc.co.uk	bump-3.js	require.min.j...	js	mis en cache	10
304	GET	an-web-asset...	bbcdotcom.is	5856195:24...	is	mis en cache	32

58 requêtes | 4,64 Mo / 326,42 Ko transférés | Terminé en : 8,60 s | DOMContentLoaded: 1,20 s | load: 8,

En-têtes | Cookies | Requête | Réponse | Délais | Trace de la pile | Sécurité

Filtrer les propriétés | curl https://weather-broker-cdn.api.bbc.co.uk/en/forecast/aggregated/5856195 > forecast.json

JSON Brut

```
{
  "type": "FeatureCollection",
  "features": [
    {
      "type": "Feature",
      "id": "5856195",
      "geometry": {
        "type": "Point",
        "coordinates": [
          -157.85833,
          21.30694
        ]
      },
      "properties": {
        "observations": [
          {
            "time": {
              "utc": "2021-09-28T05:00:00Z",
              "timezone": "Pacific/Honolulu",
              "offset": "-10:00"
            },
            "temperature": {
              "c": 26,
              "f": 79
            },
            "windDirection": {
              "direction": "NE",
              "description": "North Easterly"
            },
            "averageWindSpeed": {
              "mph": 11,
              "kph": 18
            },
            "maxWindGustSpeed": {
              "mph": null,
              "kph": null
            }
          }
        ]
      }
    }
  ]
}
```


Fetch the raw data and version it

```
$ curl https://weather-broker-cdn.api.bbci.co.uk/en/forecast/
aggregated/5856195 > forecast.json
```

```
$ du -h forecast.json
236K
```

```
$ git add forecast.json
$ git commit --message "$(date)" --allow-empty
$ git push
```

Use GitHub Actions to run periodically

```
name: update-honolulu-forecast
```

```
on:
```

```
  schedule:
```

```
    - cron: "30 * * * *" # every hour at the 30 minute mark
```

```
jobs:
```

```
  ubuntu:
```

```
    runs-on: ubuntu-latest
```

```
    steps:
```

```
      - uses: actions/checkout@v2
```

```
        with:
```

```
          fetch-depth: 0
```

```
      - run: |
```

```
        git config user.name github-actions
```

```
        git config user.email github-actions@github.com
```

```
      - run: |
```

```
        curl https://weather-broker-cdn.api.bbci.co.uk/en/forecast/aggregated/5856195 > forecast.json
```

```
        git commit -m "$(date)" --allow-empty
```

```
        git push
```

Only the relevant differences are stored

Showing 1 changed file with 113 additions and 134 deletions.

247 forecasts.json			
...	...	@@ -1,125 +1,104 @@	
1	1	[
2	2	{	
3	3	"issueDate": "2021-07-09T11:00:00-10:00",	
4	-	"lastUpdated": "2021-07-09T20:23:42.526-10:00",	
	4	+	"lastUpdated": "2021-07-09T21:24:22.447-10:00",
5	5	"reports": [
6	6	{	
7	-	"extendedWeatherType": 0,	
	7	+	"extendedWeatherType": 38,
8	8	"feelsLikeTemperatureC": 31,	
9	-	"feelsLikeTemperatureF": 88,	
	9	+	"feelsLikeTemperatureF": 89,
10	10	"gustSpeedKph": 26,	
11	11	"gustSpeedMph": 16,	

Is it space efficient? Yes, it is.

- I collected the Honolulu weather forecast every hour for ~2800 hours (for free)
- The data weighs ~15 megabytes
- Storing the raw data each hour would have required ~652 megabytes
- That's a **x43 reduction** in storage requirements

But how do you analyse the data?

- The data is stored in a `.git` directory
- You can use git to time travel over the data ✨
- e.g. “*What was the forecast n days ago?*”
- This can be very powerful!

1) Dump the data to SQLite

```
import sqlite3
import git
import pandas as pd

def load_forecasts_from_commit(commit: git.Commit) → pd.DataFrame:
    ...

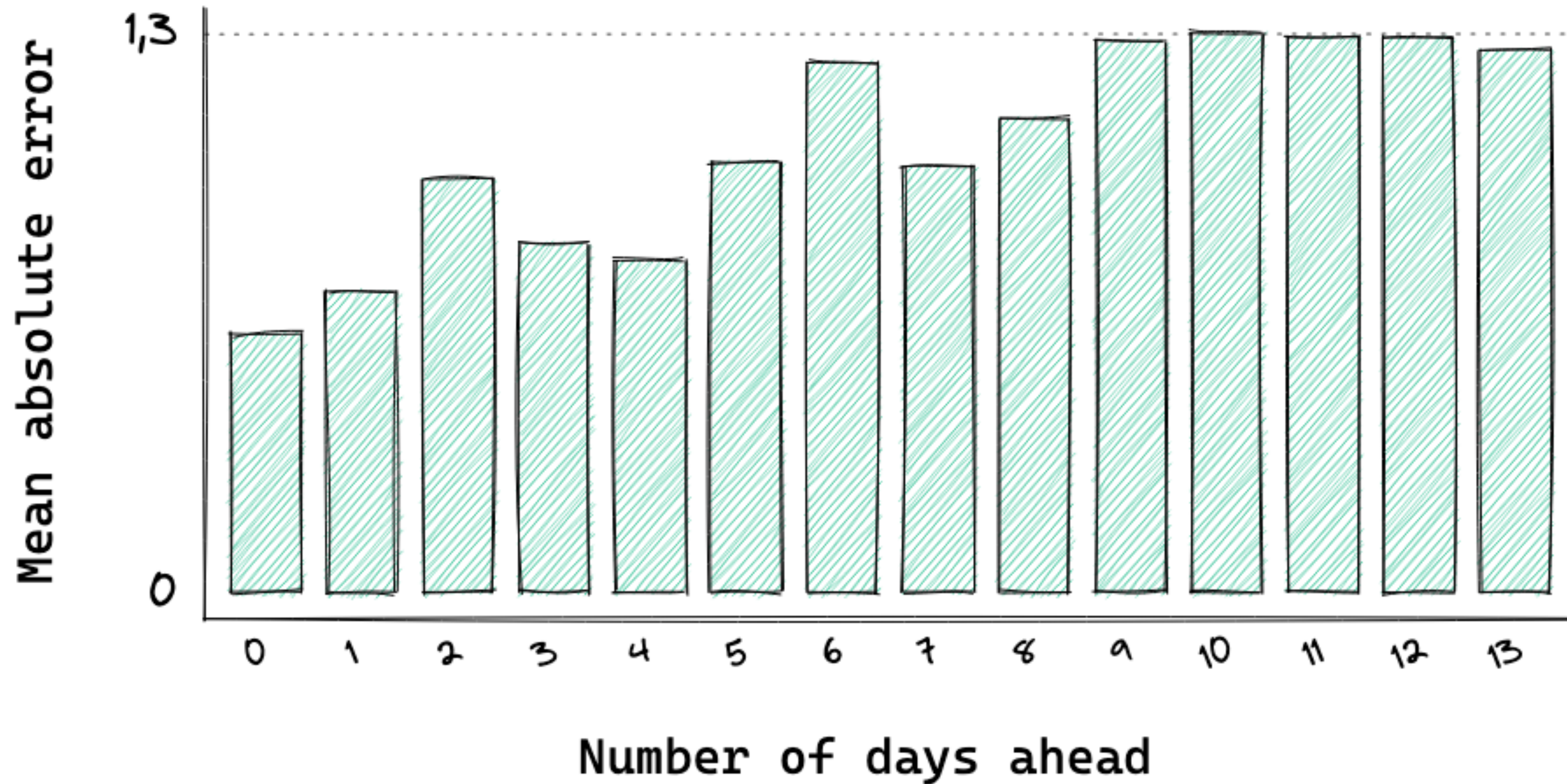
sqlite_conn = sqlite3.connect("honolulu_weather.sqlite")
commits = git.Repo(".").iter_commits()

for commit in commits:
    forecasts = load_forecasts_from_commit(commit)
    forecasts.to_sql("forecasts", con=sqlite_conn, if_exists="append")
```

2) Run a SQL query

```
WITH deltas AS (  
    SELECT  
        ROUND(JulianDay(f.at) - JulianDay(f.issued_at)) AS days_ahead,  
        o.celsius - f.celsius AS celsius  
    FROM observations o, forecasts f  
    WHERE o.at = f.at  
)  
  
SELECT  
    days_ahead,  
    AVG(ABS(celsius)) AS mae_celsius  
FROM deltas  
GROUP BY days_ahead;
```


3) Copy/paste into Excalidraw 🧙



Advantages

- Little effort required
- Storage size is minimal
- Application agnostic
- Data is stored on GitHub, making it easy to share
- GitHub Actions' free tier is permissive

👁👁 Further reading

- github.com/MaxHalford/bbc-weather-honolulu
- I didn't invent this, but Simon Willison might have
- Check out the discussion on Hacker News
- Datasette is worth checking out
- You could also use Zapier for automation
- Sensible git diffs for JSON file

Thank
You